Lehrhilfe

Bürocomputer Softwaretechnologie

VEB ROBOTRON
ANLAGENBAU
SCHULUNGSZENTRUM

robotron 1405

VEB Robotron-Anlagenbau
Schulungszentrum

L E H R H I L F E

Softwaretechnologie

BC A 5110, A5120, A 5130

Ag 706/358/84 III/8/1 888

1. Auflage

Als Manuskript ohne Lektorierung gedruckt.

Ausgearbeitet: Autorenkollektiv
des VEB Robotron-Anlagenbau
Schulungszentrum

Dieses Material wurde ausschließlich für Schulungszwecke geschaffen und unterliegt nicht dem Änderungsdienst. Nachdruck oder Vervielfältigung, auch auszugsweise, ist nur mit vorheriger schriftlicher Zustimmung des Herausgebers statthaft. Die Weitergabe oder der Weiterverkauf an Dritte ist nicht erlaubt.
Diese Lehrhilfe wurde mit dem Elektronischen Schreibsystem ESS A 5310 erfaßt, korrigiert, ausgedruckt und steht dem Schulungszentrum als Textkonserve zur Verfügung.

© VEB Kombinat Robotron 1984

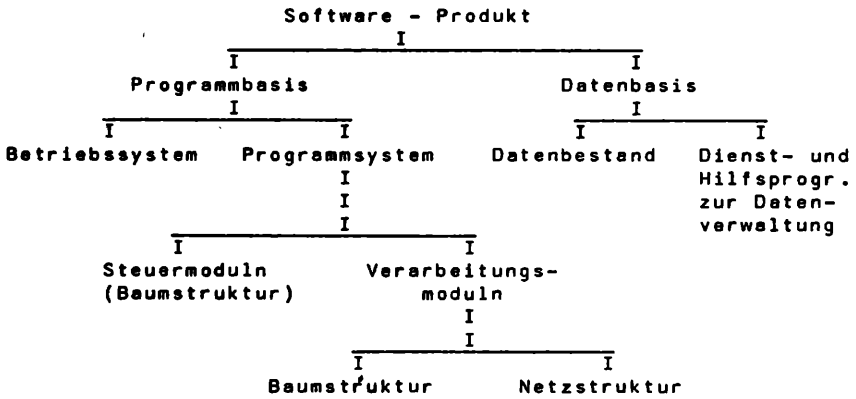
I N H A L T S V E R Z E I C H N I S

		Seite
1.	Ziele und Methoden der Software-Technologie	4
1.1.	Software-Produktion	4
1.2.	Technologischer Prozeß	6
2.	Spezifizieren der Aufgaben	10
2.1.	Arbeitsauftrag	10
2.2.	Aufgabendekomposition	10
2.3.	Teilaufgabenbeschreibung	10
2.4.	HIPO-Technik	11
3.	Konstruktion der Datenbasis	13
3.1.	Logische Datenstrukturen	13
3.2.	Physische Datenstrukturen	14
3.3.	Dateiformen und Datenmanipulation	15
3.4.	Datentabellentechnik	20
4.	Datenorientierte Programmkonstruktion	22
4.1.	Typisierte Programmrahmen	22
4.2.	Programmskelettkonstruktion	26
5.	Funktionsorientierte Programmkonstruktion	29
5.1.	Funktionale Dekomposition	29
5.2.	Logische Bausteine als Strukturprogramm	29
5.3.	Entscheidungstabellen - Ablauftabellen	31
6.	Software-Entwicklungssystem (SOES)	33
6.1.	Konzeption eines SOES	33
6.2.	Technologische Werkzeuge im SIOS 1526	35
6.3.	PSEUDO - Softwareentwicklungswerkzeug	38
6.4.	ESEM - Softwareentwicklungswerkzeug	43
6.5.	SEP am ESS A 5310 mit TEXT20	45

1. Ziele und Methoden der Software-Technologie

1.1. Software-Produktion

Software als Produkt
- Lösung einer Datenverarbeitungsaufgabe in Form von Dokumentation, Programmen, Arbeitsvorschriften, Datenträgern.
- Produktionsmittel zur Erzeugung benötigter Informationen für Leitung und Steuerung von Prozessen.
- Arbeitsergebnis mit hohem Nutzeffekt und erforderlichen Gebrauchseigenschaften.
- Wissenschaftlich-technische Leistung, welche auch durch Dritte genutzt werden kann.



Datenbestand
= Gesamtheit der strukturierten Daten in einer Aufgabe

Datei
= Teil des Datenbestandes in einer definierten Organisationsform

Programmsystem
= Gesamtheit der Programme in einer Aufgabe, welche vom Betriebssystem verwaltet und vom Bediener zweckmäßig aktiviert werden.

Programm
= Hierarchie von Moduln, die eine Teilaufgabe realisiert und durch das Betriebssystem selbsttätig startbar ist.
(Topmodul - Steuermodul - Verarbeitungsmodul)

Modul
= kleinste selbständige Programmeinheit, die eine abgegrenzte Funktion realisiert. Hauptteile sind: Modulinterface, Datenteil, Verarbeitungsteil, Dokumentationsteil.

Qualitätsmerkmale des Software-Produkts TGL 36800

Qualität
= Gesamtheit der Erzeugniseigenschaften, die den Grad der Eignung von Software (Gebrauchswert) für den vorgesehenen Verwendungszweck bestimmen.

Qualitätsmerkmale:

- Korrektheit	. Vollständigkleit der Lösung (Rationalität) . Übereinstimmung zwischen Auftragsspezifikation, Programm, Dokumentation und Arbeitsvorschriften
- Effizienz (Effiktivität)	. Grad der Ressourcenauslastung . Grad des Aufwandes im Dauerbetrieb (Kosten, Energie, Material) . Möglichkeit nachträglicher Leistungsverbesserung
- Wiederverwendbarkeit (Flexibilität)	. Grad der Nachnutzbarkeit . Grad der Kombination mit anderer Software (Integrationsfähigkeit)
- Verträglichkeit (Universalität)	. Grad der Standardisierung = Anteil typisierter Elemente . Grad des Umfangs der Anpassungsarbeiten
- Verständlichkeit (Transparenz)	. Grad der Überschaubarkeit . Grad des Einarbeitungsaufwandes
- Zerlegbarkeit (Lokalisierung)	. Anpaßbarkeit an ein Betriebssystem . Ortung von Störursachen . Logische Geschlossenheit der Elemente
- Überprüfbarkeit (Normalisierung)	. Verfügbarkeit geeigneter Testprogramme . Grad der Testdatenbereitstellung . Vollständigkeit der Testvorlagen

1.2. Technologischer Prozeß

Der Entwicklungsvorgang eines Software-Produkts

ist ein planmäßiger, (1)
rationell organisierter Arbeitsprozeß, (2)
in dessen zielgerichtetem Verlauf (3)
wissenschaftlich-technische Leistungen (4)
erbracht werden (arbeitsteilig).

Intensivierung des Arbeitsprozesses
trägt zur Erhöhung der Leistungsfähigkeit des Entwicklerkollektivs wesentlich bei

```
Arbeitsauftrag
      I
      I
Arbeitsprozeß  ===  SE - Technologie
      I                 .Werkzeuge
      I                 .Hilfsmittel
      I                 .Prüfmittel
Softwareprodukt
```

Werkzeuge, Hilfsmittel, Prüfmittel (W, V, P)

= sind Mittel, welche auf Grund ihrer Eigenschaften vom Menschen benützt werden, um einen bestimmten Zweck zielgerichtet und erfolgreich in kurzer Zeit zu realisieren.

Der Technologische Prozeß

= der Teil des Arbeitsprozesses, der unmittelbar mit dessen zielgerichteten Verlauf verbunden ist.

Die Technologie

= die Widerspiegelung eines technologischen Prozesses mittels Modell als Ergebnis der wissenschaftlich-technischen Analyse und als vorweggenommenes Abbild des konkreten Prozesses.

Die Aufgaben im technologischen Prozeß

Allgemeine Technologie	SE-Technologie
Aufgaben zur Variantenbetrachtung:	
- Variantenfindung	- Varianten erarbeiten
- Variantenbewertung	- Zuordnen der Parameter, Kriterien
- Variantenvorentscheidung	- Verteidigung, Pflichtenheft
- Variantenvergleich	- Varianten vergleichen
- Variantenauswahl	- Konzeption, Programmentwurf
Aufgaben für den Gesamtprozeß:	
- Prinzipielle Entscheidung zum Entwicklungsverfahren	- Prinzipielle Entscheidung zur Anwendung der SE-Technologie
- Bestimmung des Ausgangszustandes	- Bestimmung der Bedingungen, Aktionen, Informationen
- Bestimmung von Prozeßelementen	- Elementarisierung nach Daten und Verarbeitungsschritten
- Bestimmen der Prozeßfolge	- Programmablauf, Datenfluß, Verarbeitungsschrittfolge
Aufgaben zur Gestaltung des Arbeitsprozesses:	
- Lagerung vor, zwischen und nach dem Arbeitsgang	- arbeitsbegleitendes Dokumentieren, SE-Bibliotheken
- Festlegung von Kontrollen	- Qualitätskontrolle, Verteidigung
Aufgaben zur Detailierung der Arbeitsgänge:	
- Gliederung in Teilaufgaben, Handgriffe, Griffelemente	- schrittweise Verfeinerung, Formalisierung, Elementaroperationen, Dekomposition
- Bestimmen der Organisation der Arbeiten	- Arbeitsteilung, Spezialisierung, Arbeitsplanung und Abrechnung
- Klassifizierung der Arbeiten	- Typisierung der Bausteine, Sekundäroperationen
- Bestimmung der Aufwandsparameter	- Bestimmung von Speicherplatz, Zugriffsdauer, Papierverbrauch u.v.m.

Phasenmodell der SE-Technologie

Entwicklungsphasen im Arbeitsprozeß sind:

1. Analysephase (Spezifizieren)
2. Entwurfphase (Konstruieren)

---

3. Implementierphase (Codieren, Programmieren)
4. Testphase (Abschließende Qualitätskontrolle und Nachbesserung)

Phasenmodell:

Schnittstelle	Phase	Arbeitergebnis
Nutzer Auftragleiter	Arbeits- auftrag	Aufgabenstellung, Arbeitsauftrag
Nutzer Analytiker	Analyse	Prozeßstruktur, Prozeßelemente Informationsarten, Anforderungen Ziele
Analytiker Konstrukteur	Entwurf	Programmlogik, Funktionen, Algorithmen, logische Daten- strukturen und Schnittstellen
Konstrukteur Programmierer	Implementieren (Fertigung)	Steuermoduln, Verarbeitungs- moduln, Dateien, Parameter
Programmierer Auftragleiter	Test Erprobung	Arbeitsvorschriften für Nutzer und Bediener, Datenträgerver- waltung, Demonstrationsbeispiel
Auftragleiter Nutzer	Fertigmeldung	Datenträger Dokumentation

Arbeitsbegleitende Tätigkeiten sind in allen Phasen:

- dokumentieren : Arbeitergebnis beschreiben
- verwalten : Leitung, Lenkung, Abrechnung, materielle Sicherstellung, Ressourcenbilanzierung, Koordinierung der Arbeiten
- kontrollieren : Qualitätssicherung, Berichterstattung, Tiefenprüfung, Verteidigung WTL

Software-Entwicklungs-Technologie (SE-Technologie)

= Rahmentechnologie (engl.: software engineering)
mit dem Ziel:
. Erhöhung der Effektivität des Arbeitsprozesses
: Senkung des Aufwands (Zeit, Kosten, Material, Energie)
: Steigerung der Arbeitsleistung an jedem Arbeitsplatz
. durch ingenieurgemäße Arbeitsweise
: Konstruktion der Lösung
: Katalogisierte typisierte Bausteine
: Mittel (Werkzeuge, Hilfsmittel, Prüfmittel)
. mit Computer-Unterstützung als Rationalisierungsmittel.

Prinzipien:
= Prinzip der Abstraktion
.Grundgedanke - Programmentwicklung solange wie möglich vom konkreten Implementieren auf BC abhalten, damit wird eine allgemein gültige Programmiervorlage geschaffen.
.Baumstrukturen für Steuermoduln bevorzugen
.Netzstrukturen für Verarbeitungsmoduln einsetzen
= Prinzip der Steuerflußdisziplinierung
.Grundgedanke - Erhöhung der Transparenz
Fernwirkungsfreies Auswechseln der Strukturelemente zwecks
: Erhöhung der Effektivität
: Beseitigen von Qualitätsmängeln
: Erneuerung von Bausteinen
.Blockkonzept - Jeder Baustein ist ein Block mit definiertem Eingang und Ausgang (blocinterface)
.Verzweigungen - logisch zum Blockeingang nur erlaubt
.Blockanschlußbild (Eingang/Ausgang) - wird mit definierten Datenstrukturen realisiert
= Prinzip der Computernutzung
.Grundgedanke - Datenerfassung erzeugt zusätzliche Fehler, was zu vermeiden ist
.Rationelle Arbeitsweise bei der Softwaregestaltung
: Problemorientierter Quelltext
mit PSEUDO, ESEM, INGE, LISG, EDITOR, TEXT, LBSV
: BASIC - Quelltext
mit Editor, Interpreter, Programmdateien
: PASCAL - Quelltext
mit Editor, Compiler, Interpreter, Bibliotheksservice
: Assembler-Quelltext
mit Übersetzer (MBAS, CASS, SASS), EDITOR, LBSV
: Dienstprogramme zur Dateimanipulation
mit INIT, FGEN, KONV, DINH, DIDI, COPD, CODP, u.a.

Entwicklungsetappen eines Programmes im BC-Dialog

1. Prosa, Text (verbale Beschreibung, logisch gegliedert)
2. Tabellen, Diagramme (Strukturen, Bausteine)
3. Pseudocode, Tabellen (Bausteinfunktion, Unterprogramm)
4. Quelltext in Programmiersprache (Datendefinitionen, Moduln)
5. Maschinencode (Makros, Phasen)
6. Dokumentation (Prosa, Diagramme, Tabellen, Quelltext, Protokolle, Testdaten)

2. Spezifizieren der Aufgaben

2.1. Arbeitsauftrag

Der Arbeitsauftrag
= vom gegenwärtigen Erkenntnisstand geprägt und enthält im allgemeinen eine betriebswirtschaftliche Aufgabe im Rahmen von Rationalisierungsvorhaben, zu deren Lösung BC eingesetzt werden sollen.

Betriebswirtschaftliche Aufgabe
= Aussageform (A,B)
.gegeben sind Bedingungen (A),
.gefordert werden Ziele (B).

Spezifizieren der Aufgabe (Auftragsspezifikation)
(A,B) := (A,-),(-,B)

2.2. Aufgabendekomposition

Aufgabendekomposition
zerlegt die Gesamtaufgabe in Teilaufgaben (A,B), dabei werden Bedingungen und Ziele präzisiert.

1. Aufgabenstrukturierung
2. Aufgabenbeschreibung

2.3. Teilaufgabenbeschreibung

Beschreibung der Strukturelemente enthält:
- Identbegriff (Name, Nr.)
- Bezeichnung (Kurzform)
- Inhaltskurzfassung (Langform)
- Aufzählung der enthaltenen Komponenten
- Beziehungen zu anderen Teilaufgaben
- Paramater, Kriterien, Strukturdiagramm

Beschreibung der Verarbeitungselemente:
- Identbegriff (Name, Nr.)
- Bezeichnung (Kurzform)
- Inhaltskurzfassung (Langform)
- Aufzählung der E/A-Informationen (Daten)
- Beschreibung der Verarbeitungsschritte
- Beziehung zu anderen Teilaufgaben
- Parameter, Kriterien, Grenzen
- Quellen, Senken, Periodizität, Vertraulichkeitsgrad

2.4. HIPO -Technik (hierarchy-input-process-output)

Grundgedanke:
- eine allgemein verständliche Dokumentation schaffen
- den Spezialisten wenig Vorschriften auferlegen
- die Symbolik der bestehenden Technologie anpassen
- die Erkenntnisse als Überblick und im Detail darstellen
- die Überschaubarkeit gewährleisten
- die ingenieurmäßige Arbeitsweise unterstützen
- unabhängig vom Industiezweig und Auftrag vereinheitlicht dokumentieren
- Kontrolle im Prinzip: walk-through, Verteidigung WTL

Notationsformen:
- Hierarchiediagramm als Gliederung des Inhalts

- Überblicksdiagramm als Beschreibung des Zusammenhangs (E, V, A)

- Detaildiagramm als Beschreibung der Einzelheiten bis zum (E, V, A) Programmentwurf

Dokumentation:
- arbeitsbegleitendes akkumulieren gültiger Blätter
  = Beschreibung aller Arbeitsergebnisse
  = Dokumentationsbibliothek
- Verwalten, zwischenlagern,
  Arbeitsvorbereitung erfolgt durch Verwendung ausgewählter Blätter als Basis für den nächsten Arbeitsgang.
- Bericht erstatten, Nutzer informieren, Bediener anleiten erfolgt durch zusammenstellen ausgewählter Blätter

Symbole:
- siehe Seite 12

Symbole für HIPO-Diagramme

Hierarchie und Funktion

Text Verweis Nr.	Strukturkomponente, Funktion, Aufgabe
Text # Verweis Nr.	Komponente an anderer Stelle beschrieben
Text Verweis Nr.	Selbständige Komponente Unterprogramm
Text	Komponente bekannt, nicht enthalten

Daten, Schnittstellenanschluß

Text	Datengruppe, Datei, Information
Text	Datenelement

Datenfluß

Daten, Datengruppe	Zeiger, Verweis
Datei, Information	Schalter, Bezugnahme
Aufruf eines Unterprogramms	Programmablauf

3. Konstruktion der Datenbasis

3.1. Logische Datenstrukturen

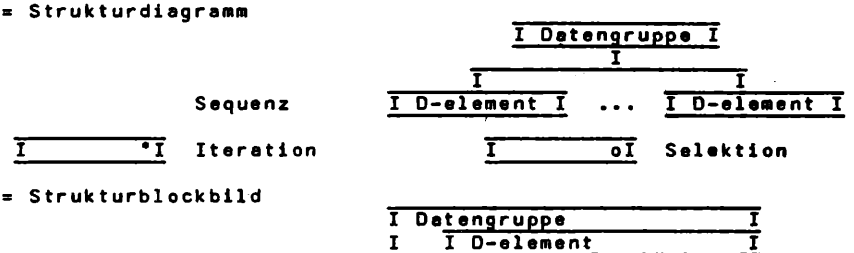
Grundgedanke:
Der Datenbestand wird mittels mathematischer Mengenlehre als Datenmenge betrachtet, nach gegebenen Kriterien analysiert und logisch gruppiert.

Datenmenge
= Ansammlung elemetarer Daten, welche in ihrer Gesamtheit zu einer Aufgabe gehören.

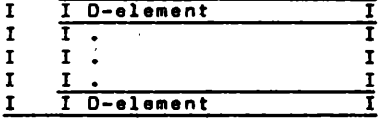
Logische Datei
= Datengruppe mit bekannter Struktur, welche mit einem Namen ansprechbar ist.

Kriterien
= Prozeßkriterien (Datengruppe je Verarbeitungsblock)
= Formalkriterien (Ordnungs-, Verarbeitungs-, Hinweisdaten)

Notationsform
= Strukturdiagramm



= Strukturblockbild



= TGL 22451 Symbole für den Datenfluß

= Datentabellen vergleiche: Datentabellentechnik

= Formalisierter Text
Datengruppe := D-element a, ..., D-element z
Klammern erlauben tiefere Strukturebenen

Methodische Anleitung zur schrittweisen Unterteilung der Datenmenge
1. Erfassen der Elemente einer zu ordnenden Datenmenge
2. Bilden von Datengruppen, bestimmen deren Unterscheidungsmerkmale
3. Überprüfen der Datengruppen auf Disjunktivität, Inklusivität
4. Definieren der Datengruppen
5. Notieren der Datengruppenstruktur

3.2. Physische Datenstrukturen

3.2.1. Diskette 8" - Standarddiskette

Diskettenaufbau:

Physische Spur	Spurbezeichnung
00	Indexspur, Satzlänge = 128 Byte
01 - 74	Datenspuren
75, 76	Ersatzspuren

Spuraufbau:
- Indexmarke - Spurbeginn
- Sektor - Kennzeichnung, Datenfeld

Datenformat, physisch (Sektor = Datenfeld):

128 Byte	=	26 Sätze je Spur
256 Byte	=	15 Sätze je Spur
512 Byte	=	8 Sätze je Spur
1024 Byte	=	4 Sätze je Spur

- bei doppelter Aufzeichnungsdichte:

128 Byte	=	44 Sätze je Spur
256 Byte	=	26 Sätze je Spur

Sektor:

Kennzeichnung	Lücke	Datenfeld	Lücke	...

Einteilung der Indexspur:

01	-	Nechlader
02 - 04	-	reserviert
05	-	Fehlerkennsatz
06	-	reserviert
07	-	Datenträgerkennsatz VOL1
08 - 26	-	Dateikennsätze HDR

Kennsätze haben die feste Länge von 80 Byte.

3.2.2. Diskette 5,25" - Minidiskette

Diskettenaufbau:

Physische Spur	Spurbezeichnung
00	Indexspur, Satzlänge = 128 Byte
01 - 37	Datenspuren
38, 39	Ersatzspuren

Spuraufbau: wie 8" - Diskette

Datenformat, physisch (Sektor = Datenfeld):

128 Byte	=	26 Sätze je Spur
256 Byte	=	15 Sätze je Spur
512 Byte	=	8 Sätze je Spur
1024 Byte	=	4 Sätze je Spur

Dateikennsatz:

extent	-	Dateibereich
BOE	-	Anfangsadresse Datenbereich
EOE	-	Endadresse Datenbereich
EOD	-	Endadresse Datenbelegung

Adresse:

Spurnummer	Sektornummer

3.3. Dateiformen und Dateimanipulationen

Dateimanipulationen

- Laden
  = aufbauen, erstellen
  = erstmalige Belegung einer Datei mit Datensätzen

- Hinzufügen
  = erweitern
  = nach dem letzten Datensatz werden weitere Sätze zwischen EOD und EOE geladen, wobei EOD neu gesetzt wird.

- Einfügen
  = erweitern, ändern
  = Datensätze werden an die logische Stelle in der Datei gespeichert, die durch die Reihenfolge des Ordnungsmerkmals gegeben ist.

- Aktualisieren
  = ändern, update
  = ein Datensatz wird gelesen und mit verändertem Inhalt auf die gleiche Adresse zurückgespeichert.

- Löschen
  = streichen, stillegen
  = ein Datensatz wird für ungültig erklärt, indem er mit H'F8' markiert wird.

- Reorganisieren mit COPD
  = wiederherstellen
  = Die Datei wird auf eine andere Diskette (Bereich) übertragen, markierte Sätze sollten nicht übernommen werden. Es wird eine Übereinstimmung zwischen logischer und physischer Organisation wieder hergestellt. Die Datei behält ihren Namen. Der erzielte Zustand entspricht dem Zustand nach dem Laden.

- Initialisieren mit INIT
  = Die Diskette wird für die Arbeit im Betriebssystem erstmalig oder erneut eingerichtet. Die Spur 0 erhält Kennsatz-Sektoren (128 Byte), die anderen Spuren erhalten Spurkennzeichnung und Datensektoren und werden geprüft. Es können höchstens zwei Ersatzspuren bei Defekten zugeordnet werden.

Dateiformen im SIOS 1526 auf Diskette
Sequentiell organisierte Datei (SD)

Die Datensätze sind lückenlos hintereinander gespeichert. Der Zugriff erfolgt immer zum nächsten Satz. Der erste und der letzte Satz sind direkt adressierbar (EOB, EOD). Die Datensätze können nach einem Ordnungsmerkmal (Schlüssel) sortiert (aufsteigende oder absteigende Reihenfolge) oder ungeordnet in der Datei stehen. Das Ordnungsmerkmal wird beim Zugriff nicht berücksichtigt.

Positionierung:

=	F	- first	- RA:=1	- Zugriff auf den 1. Satz
=	L	- last	- RA:=n	- Zugriff auf den letzten Satz
=	N	- next	- RA:=RA +1	- Zugriff vorwärts nächster Satz
=	B	- back	- RA:=RA	- Zugriff rückwärts nächster Satz
=	E	- extent	- RA:=n +1	- Zugriff nach dem letzten Satz

Dateimanipulation:
- Laden der Datei kann mit einem Programm erfolgen, welches mit INGE generiert wurde.
- Hinzufügen neuer Datensätze erfolgt nach dem letzten Satz von EOD bis EOE, wobei EOD neu gesetzt wird.
- Einfügen von Datensätzen in die sortierte Reihenfolge des Ordnungsmerkmals ist nicht möglich, deshalb sollten diese Sätze hinzugefügt werden und dann muß mit SORD (Ordnungsmerkmal max. 32 Stellen) ein Neuordnen erfolgen.
- Aktualisieren erfolgt mit der Positionierung B im nächsten Zugriff (WRITE B), damit erfolgt ein erneuter Zugriff zur selben Satzadresse.
- Löschen = erzeugen markierter Sätze (H'F8'). Der Inhalt des Datensatzes bleibt erhalten, weil die Markierung außerhalb vor dem Datensatz gespeichert wird.
  Markierte Sätze werden beim Zugriff übergangen.
- Reorganisieren kann mit COPD (auf eine andere Diskette kopieren) erfolgen, wobei markierte Sätze nicht zu übernehmen sind.

Satzformate : ungeblockt, geblockt
Mehrdatenträgerbetrieb
= mit einem oder mehreren Laufwerken möglich

Der direkte Zugriff kann ermöglicht werden, denn bei "schnellen Suchen" wird die Datei solange sequentiell durchsucht, bis ein Suchmerkmal den relevanten Schlüssel gefunden hat. Die zugehörige Satzadresse wird vom Betriebssystem bereitgestellt. Die Startposition für die schnelle Suche kann vorgegeben werden, jedoch nicht bei geblockten Sätzen.

Dateiformen im SIOS 1526 auf Diskette

Direktadressiert organisierte Datei (DA)

Die Datensätze sind getrennt verteilt gespeichert. Sie stehen nicht lückenlos hintereinander, woraus eine erhöhte Bereichsgröße der Datei resultiert. Auf die Datei kann sequentiell oder direkt zugegriffen werden.

Positionierung:
Zwischen Ordnungsmerkmal (numerischer Schlüssel) und Satzadresse besteht ein direkter Zusammenhang. Für jede mögliche Schlüssel-Nr. wird ein Satz gespeichert, d.h. belegte Schlüssel-Nr. ergeben Datensätze, nicht genutzte Schlüssel-Nr. ergeben Leersätze. Die Leersätze sollten markiert werden. Bei sequentiellem Zugriff hat die Datei eine aufsteigend geordnete Satzfolge.

Dateimanipulation:
- Laden erfolgt in eine sortierte sequentielle Datei, wozu ein Programm mit INGE erzeugt werden kann. Danach wird mit KONV die direktadressierte Datei geladen, indem die Datensätze auf die zugeordneten Satzadressen verteilt werden. Nicht belegte Schlüssel-Nr. ergeben markierte Leersätze. Doppelte Schlüssel-Nr. führt zum Abbruch wegen Fehler.
- Hinzufügen neuer Datensätze erfolgt am Ende der Datei zwischen EOD und EOE, wobei EOD neu gesetzt wird.
- Einfügen neuer Datensätze erfordert eine Aktion, welche die Markierung löscht, dann kann der Satz an der zugeordneten Adresse gespeichert werden.
- Aktualisieren im Satz wird mit einem zweiten Zugriff zur selben Adresse realisiert. Vorteilhaft ist der direkte Zugriff (WRITE B).
- Löschen: Der Inhalt des Satzes bleibt erhalten, weil die Markierung außerhalb vor dem Satz gespeichert ist. Markierte Sätze werden beim Zugriff übergangen.
- Reorganisieren kann mit COPD durch Kopieren auf eine andere Diskette erfolgen, wobei markierte Sätze nicht zu übernehmen sind (Markierung für Leersätze und gelöschte Sätze vorhanden). Wiederherstellen der Datei auf alter vorgerichteter Diskette mittels KONV möglich.

Satzformate : ungeblockt,
geblockt nur bei Blockadresse sinnvoll

Mehrdatenträgerbetrieb :
= ist mit mehreren Laufwerken möglich.

Dateiform im SIOS auf Diskette
Indexadressiert organisierte sequentielle Datei (IS)

Die Datensätze stehen aufsteigend nach Ordnungsmerkmal sortiert lückenlos hintereinander. Der Zugriff erfolgt entweder sequentiell auf den nächsten Satz oder indexadressiert direkt zur Satzadresse.

Positionisierung:
Zwischen dem Ordnungsmerkmal (Schlüssel) und der Satzadresse besteht keine logische Relation, deshalb kann der Schlüssel numerisch, alphanumerisch, alphabetisch, mit großen Belegungslücken und Doppelbelegungen ausgelegt sein, darf jedoch H'00' und H'FF' nicht enthalten. Der indexadressierte Zugriff erfolgt über den Schlüssel. Vom Betriebssystem wird beim ersten Befehl READK eine Indextabelle im bereitzustellenden RAM-Bereich erstellt, welche eine Zuordnung von Spuradresse und höchsten Schlüsselwert der Spur enthält.
Jeder folgende Befehl READK durchsucht die Indextabelle bis zum Schlüssel kleiner gleich Schlüsselwert, positioniert mit der gelesenen Spuradresse die Lese/ Schreibeinrichtung zur Diskette und vergleicht sequentiell in der Spur die Datensätze bis zur Schlüsselgleichheit. Die ermittelte Satzadresse wird vom Betriebssystem bereitgestellt. Besitzen mehrere Datensätze den gleichen Schlüssel, so wird der Satz mit der niedrigsten Satzadresse bereitgestellt. Zu den anderen Sätzen kann mit READ POS zugegriffen werden.

Dateimanipulation:
- Laden der Datensätze erfolgt in eine sortierte sequentielle Datei, wozu ein Programm mit INGE erzeugt werden kann.
- Hinzufügen neuer Datensätze erfolgt am Ende der Datei zwischen EOD und EOE in sortierter Folge, wobei EOD neu gesetzt wird. Zusätzlich wird vom Betriebssystem die Verwaltung eines Folgebereiches unterstützt. Wird der gewünschte Datensatz nicht indexadressiert gefunden, so erfolgt ein schnelles Suchen im Folgebereich.
- Einfügen neuer Datensätze erfolgt wie das Hinzufügen in den Folgebereich. Danach ist die Datei mit SORD zu sortieren.
- Aktualisieren im Datensatz wird mit einen weiteren Zugriff zur Satzadresse (WRITE B) realisiert.
- Löschen: Der Inhalt des Datensatzes bleibt erhalten, weil die Markierung außerhalb vor dem Datensatz gespeichert wird. Markierte Sätze werden beim Zugriff übergangen.
- Reorganisieren kann mit COPD erfolgen, wobei markierte Sätze nicht mit auf die andere Diskette zu übernehmen sind.

Satzformate : ungeblockt

Mehrdatenträgerbetrieb :
= mit mehreren Laufwerken möglich

Dateiformen im SIOS 1526 auf Diskette

Programmbibliotheken

Eine Bibliothek besteht aus zwei Bereichen:

VERZEICHNIS
DATENTEIL (Bücher)

Verzeichnis:
- belegt 1. Spur mit 18 Sektoren
- 1. Sektor = Benutzerkennsatz
- 17 Sektoren = Verzeichnis
- Verzeichniseintragungen (max. 255)
  in Form: /Buchname / Adresse / Typ, Länge /
  5 Byte 2 Byte 1 Byte

Datenteil:
- SL (SOURCELB)
  .Belegung je Viertelsektor (d.h. Satzlänge 128 +1)
  .1.Byte = Zeilenbeginnzeichen H'1D'
  Endekennzeichen H'0D'
  .2.Byte = Buchindex H'01' = BI = H'FF'
  Zählnummer im Verzeichnis
  Gelöschte Quellzeilen enthalten H'00'

- RL, PL (RELOCLB, PHASELB)
  .Belegung ab Sektoradresse fortlaufend bis Endekennzeichen
  .Belegung endet immer an Sektorengrenze

Buch:
- Folge von Programmanweisungen (Befehlen), die lückenlos hintereinander stehen. Diese Zeichenkette wird als Block behandelt und mit einem Namen (5 Byte) markiert. Der Name und die Adresse kommen in das Verzeichnis, während das Programm als Buch im Datenteil gespeichert wird. Programmanweisungen enthalten Großbuchstaben, Ziffern, Sonderzeichen.

3.4. Datentabellentechnik (DTT)

Grundgedanke:
- Tabellen haben weniger Zeichenaufwand als Diagramme, sind aber gegenüber Text eine übersichtlichere Darstellungsform.
- Datentabellen haben eine typisierte Grundform, zur geordneten Darstellung der Daten und ihrer Relationen.
- Datentabellen unterstützen die dialogorientierte Arbeitsweise am Computer, weil sie zeilenweise gespeichert werden können.
- Datentabellentechnik stellt spezielle Formen als Vorlagen bereit, welche leicht der Arbeit angepasst werden können.

Notation:

Zuordnungselemente							
Datenelemente		Z1		Z2	...	Zn	
D-element 1		X		2		A	
...				1		V	
O-element m		X		3		E	

Anzeiger: relevante Relation - X
Priorität der Gruppierung - 1, 2, 3
Kriterien - A, V, E

Empfohlen wird:
Anzeiger und ihre Bedeutung zu standardisieren

Methodische Anleitung:
1. Datentabellen in einfacher Form gestalten
2. Datenelemente - Leiste ist als Vorgabe mehrfach verwendbar
3. Datentabellen kombinieren, um die Aussage zu erhöhen
4. Datentabellen dokumentieren sich selbst

Einfache Datentabellen:

Datengruppe				
Daten				

Information				
Daten				

Datei				
Daten				

Satzart				
Daten				

Attribute				
Daten				

Teilaufgabe/Funktion				
Daten				

LED, LPD, LAD				
Daten				

Datengruppen				
Ordnungsdaten				

Verwendung				
Daten				

Erweiterte Datentabellen:

Verwendung				E1		E2		V		A		AE
Daten												

E/A - Tabelle	E		A	
Daten	Quelle		Senke	

Komplexe Datentabellen:

E/A -Tabelle		Eingaben						Ausgaben				
Datengruppe												
Daten												

Prozeßablauf	E		V1		AE		V2		A1		V3		A3	
Komponenten														
Daten														

AVE -Tabelle	Ausgabe						Überblick						Eingabe			
Komponente																
Daten	A1		A2		A		V		E		E1		E2	

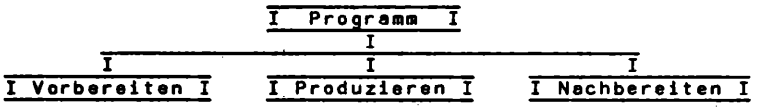
AVE-EVA-Tabelle	A		V		E		E1		E2		V1		AE		V2		A1		A2	
Komponente																				
Daten																				

Varianten			1				I				2				I				3	
Datengruppe	G1		G2		G3		G1		G2		G3		G1		G2		G3			
Ordnungsdaten																				

4. Datenorientierte Programmkonstruktion

4.1. Typisierte Programmrahmen

Grundsätzliche Programmstruktur

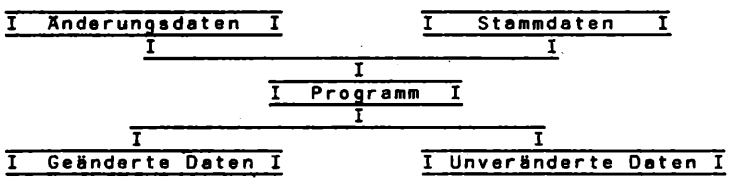


Blocksequenz:

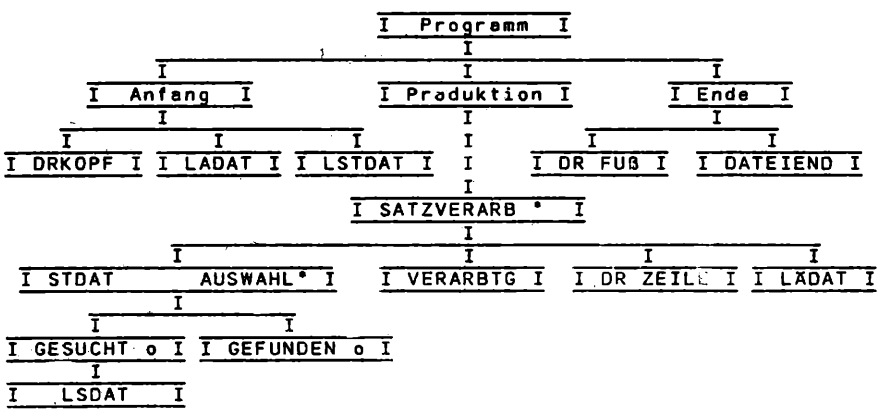
1. Block: Programmbeschreibung
   z.B. Name, Datum, u.a.
2. Block: Beschreibung der Programmumgebung
   z.B. Typ BC, Betriebssystem, Datenträger, Schalterstellungen u.a.
3. Block: Dateidefinition
   z.B. Dateibeschreibung, Arbeitsbereiche, Datenstruktur u.a.
4. Block: Programmablauf mit OP, PP, CP
   z.B. Hauptprogramm = Skellett des Programmes
5. Block: Routinen (RP)
   = Programmbausteine
   z.B. E/A-Routinen, Module
6. Block: Beschreibung nach Programmende
   z.B. Bedienhinweise, Dokumentationsteil u.a.

Programmstrukturen

- Programmstruktur für datengetriebene Verarbeitung



Methode: Änderungsfolge = steuerndes Element



Programmstrukturen

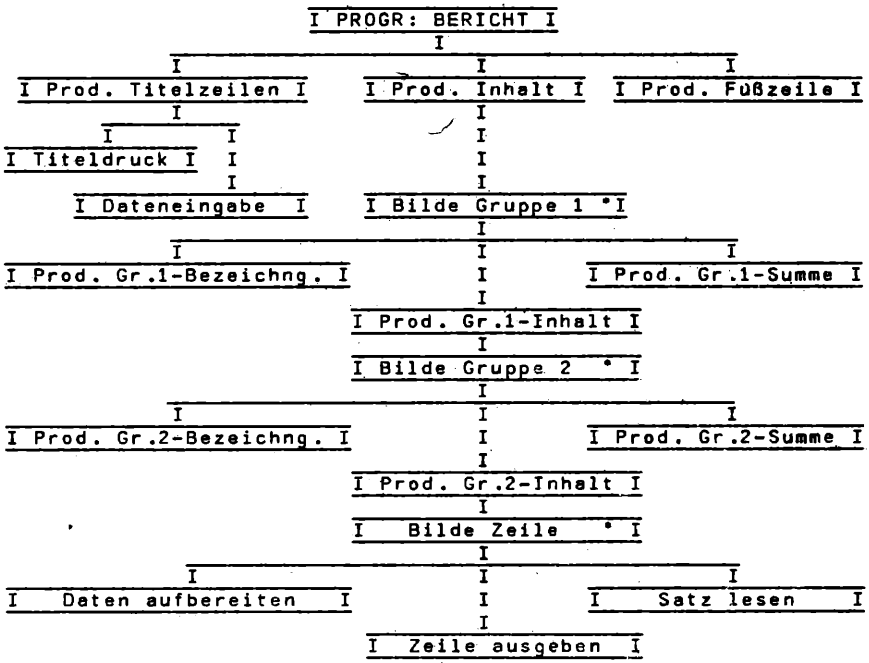
- Programmsystem ohne Programmstruktur = Programm
  = Struktur wie in 4.1.1. dargestellt

- Programmsystem mit Programmstruktur
  Jedes Programm ist ein untergeordnetes Programm, welches vom Hauptprogramm aufgerufen wird. Es enthält den 3. - 5. Block der Programmstruktur (4.1.1.). Von steuernden Daten abhängig wird als Programm eine Folge von Phasen aus einer Bibliothek abgearbeitet (CALL-Anweisungen).

  Überlagerungsstruktur
  = die aktive Phase veranlaßt das Betriebssystem die neue Phase in den gleichen HS-Bereich zu laden (d.h. die aktive Phase zu überschreiben) und ihr die Steuerung zu übergeben.
  z.B. ab AAWA oder aaaa
  I
  Anfangsadresse
  Anwenderbereich

- Programmstruktur für Gruppenverarbeitung

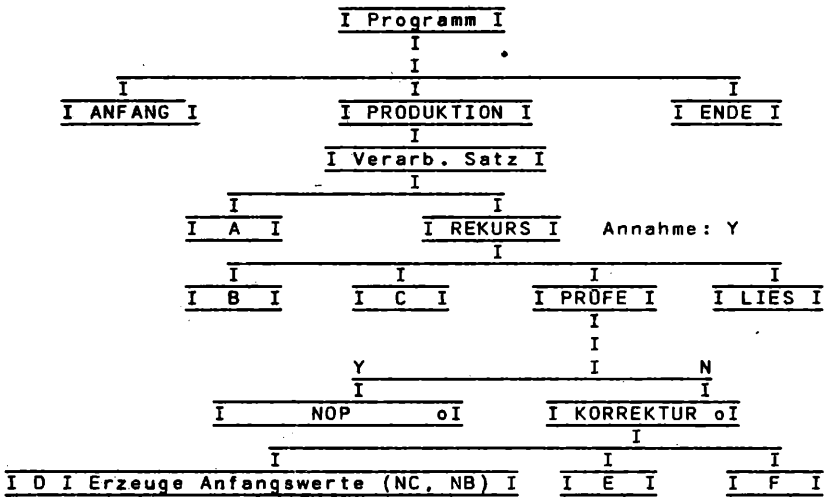
  z.B. Drucken einer Tabelle mit 2 Ordnungskriterien



Beispiel mit grundsätzlicher Struktur des Programms
(normierte Programmierung)

Programmstruktur
für rekursive Verarbeitung (backtracking)

. Rekursion
Die Richtigkeit der Verarbeitung kann erst nach ihrer Ausführung geprüft werden, deshalb ist im Fehlerfall zu bekannten Anfangswerten zurückzukehren und die Verarbeitung von dort fortzusetzen.
Die Ausführung beginnt an der Entscheidungsstelle mit einer Annahme.

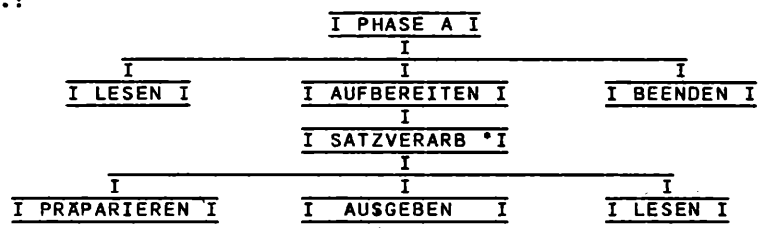


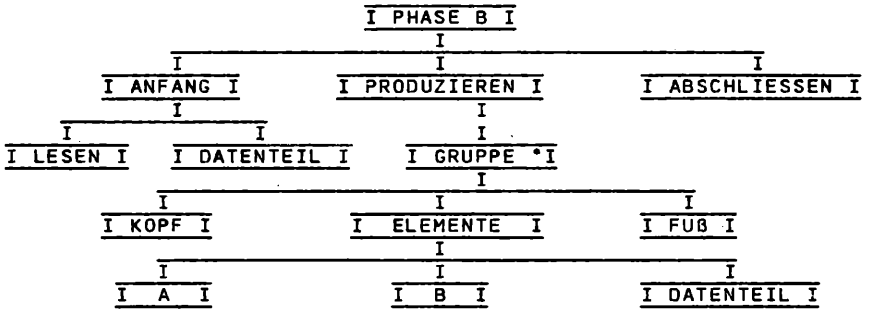
Programmstruktursubstitution

E- Daten	Struktur PHASE A	
	Datengruppe der LPD im HS	Umkehren von PHASE A nach PHASE B = Inversion
	Struktur PHASE B	A- Daten

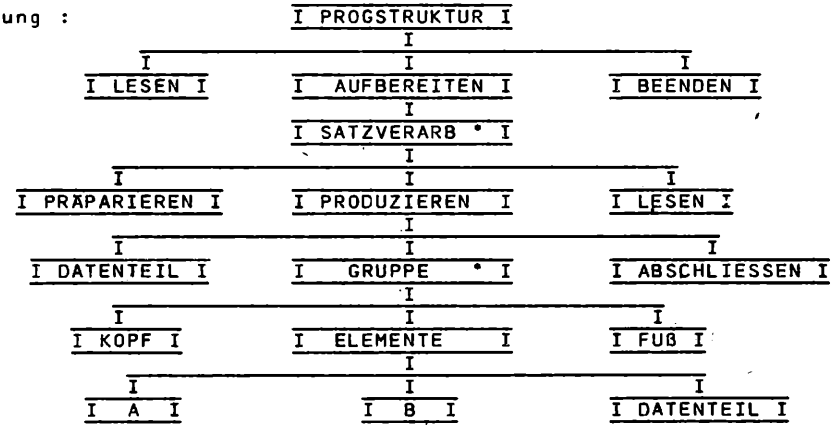
. Eine Verkettung zweier Strukturen erfolgt derart, daß die 2. Struktur an der Komponente der 1. Struktur eingeordnet wird, welche verarbeitungswürdige Daten für die 2. Struktur im HS bereithält.
Sekundäroperationen (Ausgeben, Lesen) werden in den vorliegenden Strukturen an der Verkettungsstelle gestrichen, damit wird eine höhere Effektivität durch Verkürzen der Laufzeit erzielt.

z.B.:





Lösung :



Anmerkung:
An Stelle von AUSGEBEN kann ein Verweiselement verwendet werden, z.B. PHASE B, PRODUZIERE
Dann muß die angepasste Struktur der PHASE B mit PHASE B, PRODUZIERE beginnen und kann getrennt notiert werden.

4.2. Programmskelettkonstruktion

Konstruktion der Programmbasis

Grundgedanke:
- Der Prozeß einer Aufgabe ist durch relevante Datenstrukturen beschrieben.
- Aus den Datenstrukturen wird eine angepasste Programmstruktur abgeleitet.
- Der Prozeß ist durch Einzeloperationen beschreibbar,
  primär = funktionelle Operation
  sekundär = Operation zum Datentransport
- Im Ergebnis kann die Programmkonstruktion mittels Pseudocode dokumentiert werden.

Methodische Anleitung der direkten Ableitung

1. Die prozeßrelevanten Datenstrukturen sind als Diagramme darzustellen.
2. Ermitteln korrespondierender Datenstrukturkomponenten (Verbindungspfeile notieren)
3. Ableiten der angepaßten Programmstruktur in Diagrammform,
   - indem ein Programmrahmen gegeben wird, in welchen die Strukturkomponenten eingeordnet das Programmskelett ergeben
4. Ermitteln der Operationen und Bedingungen,
   - indem elementare Operationen zuerst festzulegen sind,
   - indem E/A-Operationen davon abhängig festzulegen und einzuordnen sind.

   Als Ergebnis entsteht eine Liste der Bedingungen, eine Liste der Operationen und eine Programmstruktur in Diagrammform.
5. Das konstruierte Programm ist in geeigneter Form als Vorgabe für das Programmieren zu notieren (Pseudocode).

Methodische Anleitung der indirekten Ableitung

1. Prozeßrelevanten Datenstrukturen als Diagramm darstellen.
2. Ermitteln fehlender Korrespondenzen zwischen den E/A-Datenstrukturen.
3. Ableiten logischer Datenstrukturen, die mit einer oder mehrerer gegebener Datenstrukturen korrespondieren.
4. Prüfen, ob Dienst- oder Hilfsprogramme anwendbar sind. Diese Auswahl zuerst festlegen.
5. Direktes Ableiten der Programmstrukturen für korrespondierende E/A-Datenstrukturen (vgl. Anleitung).
6. Gestalten der Verkettungselemente für die Programmstrukturen.
7. Notieren der Vorgabe für das Programmieren.

Pseudocode

Grundgedanke:
- Die Textverarbeitung ist am besten am BC unterstützt, z.B. TEXT, EDIT
- Mit wenig Vorschriften wird ein in Umgangssprache geschriebener Baustein in eine erste einfache formalisierte Darstellung überführt.
- Die Darstellung bleibt allgemeinverständlich.
- Zur Strukturdarstellung kann der Zeilenanfang nach rechts eingerückt werden.
- Die Formalelemente werden als 'Schlüsselworte' vereinbart.

Notation:

```
I Schlüsselwort I Kommentar I
```

Jede Aussage beginnt mit einem Schlüsselwort am Zeilenanfang.

Schlüsselworte:

```
DATA       Datenblock
ACTION     Verarbeitungsblock
SEQU       Aktionsfolge
ITER       Wiederholung
               WHILE  solange
SEL        Auswahl
               IF     WENN
               THEN   DANN
               CASE   FALL
               ELSE   SONST
               OR     SONST
END....    Ende des Konstrukts
```

Methodische Anleitung:
1. Jeder Satz im Umgangstext wird mit einem Schlüsselwort begonnen.
2. Eventuell muß der Satz zerlegt werden.
3. Diese Aufbereitung ist solange zu wiederholen, bis die Kommentare programmierfähig dargestellt werden können.

Vorzugsworte zur Beschreibung von Funktion, Verarbeitung, Prozeß

Empfehlenswert ist, in einer Verarbeitungsanweisung mit einem Verb im Infinitiv beginnend die Funktion zu beschreiben, was zu tun ist. Dazu können Worte verwendet werden, wie:

abarbeiten
ableiten
abschließen
ändern
anlegen
anzeigen
aufbauen
aufbereiten
aufsuchen
ausführen
ausgeben
auswählen

beenden
beginnen
behandeln
benutzen
berechnen
bereitstellen
beschreiben
bestimmen
bilden

codieren

drucken
durchführen
durchsuchen

einfügen
einlesen
einordnen
entscheiden
entwickeln
ergänzen
erhalten
erkennen
eröffnen

errechnen
erstellen
erweitern
erzeugen

finden
formatisieren

halten
herausziehen
hinzufügen

initialisieren

konvertieren
kopieren

laden
lesen
löschen
lokalisieren

machen
mischen
markieren

ordnen

pflegen
planen
positionieren
prüfen

reorganisieren
recherchieren

schalten
schreiben

selektieren
setzen
signalisieren
sortieren
speichern
spezifizieren
starten
steuern
suchen

transformieren
tun, bis daß
tun, solange wie

überprüfen
übersetzen
übertragen
überwachen
untersuchen

veranlassen
verarbeiten
verbinden
verdichten
vermindern
verschlüsseln
vervielfachen

warten
wiederholen, von...bis
wiederholen, solange wie

zerlegen
zuordnen
zurückführen
zusammenfassen
zwischenspeichern

5. Funktionsorientierte Programmkonstruktion

5.1. Funktionale Dekomposition

Jede Aufgabe für eine BC-Entwicklung wird inhaltlich durch eine Funktion bestimmt. Im allgemeinen ist die Funktion für einen Programmbaustein zu komplex, deshalb muß diese zerlegt werden. Die Funktion wird so in Teilfunktionen zerlegt, daß die gebildeten Teilfunktionen in der dazugehörigen Struktur in jeder Ebene die Funktion vollständig repräsentieren.
Die Dekomposition wird solange fortgesetzt, bis Teilfunktionen entstehen, die einem Programmbaustein entsprechen.
An den Schnittstellen sind die Daten zu beschreiben. Die Teilfunktion selbst ist zu formalisieren und als Modell darzustellen.

5.2. Logische Bausteine als Struktogramm

Grundgedanke:
- Blockbilddarstellung ist die einfachste Diagrammform
- Konzentrierte Darstellung auf einem Blatt, d.h. keine Pfeile verwenden
- Kann deshalb am Bildschirm gute Aussagequalität erreichen
- Optimales Verhältnis zwischen Bild und Text
- Nur Grundstrukturen sind darstellbar

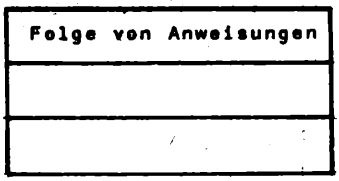
Notation:

```
NAME:
_____________________________________________________
I Deklarationen                                     I
I                                                   I
I                                                   I
I                                                   I
I                                                   I
I=====================================================I
I Anweisungen                                       I
I                                                   I
I                                                   I
I                                                   I
I                                                   
```

Notation der Programmelemente (Anweisungsteil)

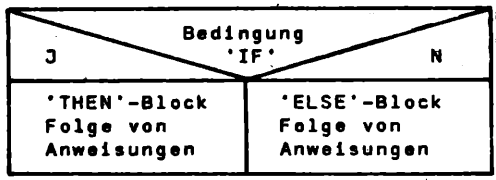
Sequenz:

NAME:



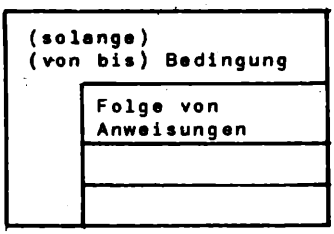
Alternative:

NAME:



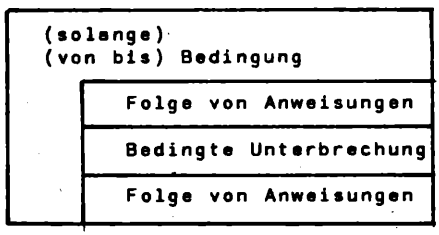
Bedingte Wiederholung:

NAME:



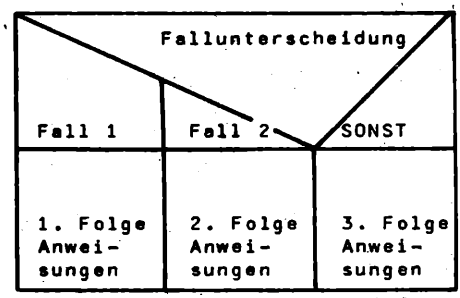
Bedingte Wiederholung mit Unterbrechung:

NAME:



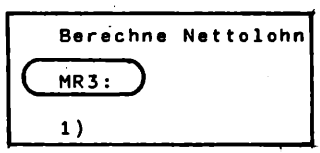
Selektion:

NAME:



Komplex:

NAME:



1) Keine Eintragungen vornehmen! Element 'MR3' wird an anderer Stelle als selbständige Einheit beschrieben.

5.3. Entscheidungstabellen - Ablauftabellen

Grundgedanke:
- In jeder Teilaufgabe/Aufgabe sind mehrere Bedingungen und davon abhängige Aktionsfolgen vorhanden.
- In einer WENN - DANN (IF-THEN) - Einteilung soll die vorteilhafte Tabellenform genutzt werden.
- Überschaubarkeit komplexer Zusammenhänge auf einem Blatt soll gewährleistet sein.
- Die Entscheidungstabelle (ET) soll von jedermann gelesen und verstanden werden.

Notation:
ET-Grundform mit horizontalen Regeln

WENN DANN
Bedingungen Aktionen

ET	I	b1	I	b2	I	b3	I...	I	bn	II	a1	I...	I.am	I	goto	I	Komment.
01	I		I		I		I	I		II		I	I	I		I	
.	I		I		I		I	I		II		I	I	I		I	
.	I		I		I		I	I		II	.	I	I	I		I	
k	I		I		I		I	I		II		I	I	I		I	

ODER UND UND

AT-Grundform mit horizontalen Regeln

Zustandsanzeige

AT	I	b1	I	b2	I	bn	I	z	II	za	I	a1	I	am	I	goto	I	Kommentar
01	I		I		I		I		II		I		I		I		I	

Bedingungen = Eingangsgrößen (b)
Aktionen = Ausgangsgrößen (a)
Zustandsgrößen (z)

Methodische Anleitung:

1. Formalisieren = Erarbeiten der Informationen, Daten, Signale und ihre Relationen.
2. Analyse = Erarbeiten der Bedingungen, Aktionen oder der Eingangsgrößen, Zustände und Ausgangsgrößen
3. Strukturierung = Erarbeiten der Bedienungskombinationen, unter denen die Prozeßelemente ablaufen
4. Entwerfen = Gestalten eines Arbeitsablaufes, Eintragen der Bedingungstexte und Aktionstexte, Setzen der Anzeiger
5. Vervollkommnen = Verbessern der ET/AT, Verknüpfen mit anderen Programmblöcken
6. Überführen in ein Programm = Ableiten eines (Pseudo)-Quelltextes
   -Regel für Regel
   -Tabellenzerlegen

Verbessern in ET / AT :
- Spaltensortieren (Bedingungen ordnen)
- Zeilensortieren (Regeln ordnen)
- Elseregel benutzen
- Zusammenfassen zweier Regeln mit gleicher Aktionsfolge, wenn sich nur Anzeigepaar unterscheidet.
- Streichen redundanter Zeilen und Spalten
- Transformieren der ET in eine andere Darstellungsform

6. Software - Entwicklungssystem (SOES)

6.1. Konzeption eines SOES

SOES haben eine Schichtenarchitektur:
- Betriebssystem (Systemkern, Systemprogramme)
- Verwaltungssystem (Datenbasis, Bibliothekskonzept)
- Softwarewerkzeuge (Methoden, Techniken, Hilfsprogramme)
- Steuerungssystem (Steuerprogramm, Steuerprozessor)
- Benutzerschnittstelle (Kommandointerpreter, Dialogführung, Nachrichtenvermittlung)

Organisation der Einführung eines SOES

1. Einweisung von 2 - 4 freiwilligen Mitarbeitern in das neue System
2. Auswählen der Werkzeuge, Techniken, Hilfsmittel Einpassen in die bestehende Technologie
3. Erstanwendung an einem streng begrenzten Auftrag mit geringen Umfang
4. Bildung einer Arbeitsgruppe und Spezialisierung der einzelnen Mitarbeiter
5. Entwerfen, implementieren, testen unter Anwendung des neuen SOES
6. Auswerten und Ausweiten auf immer mehr Mitarbeiter

Merkmale von SOES

Merkmale	Merkmalswerte
1. Progammiersprachen-bindung	- Sprachbezogen - Nichtsprachgebunden
2. Art der Software-produkte-	- Systemprogramme - Komerzielle Software - Steuerungssoftware - spezielle Software
3. Bindung an Hardware / BS	- Systemspezifisch - BS - spezifisch - portabel
4. Gerätearchitektur des Softwareentwick-lungssystem	- Inselbetrieb - Rechnernetze lokal - Rechnernetze mit DFV
5. Betriebsart	- Dialogbetrieb - Stapelbetrieb - ...
6. Struktur der Software	- Modular - Modular hierarchisch - ...
7. Modulkonzepte	- Moduln mit Datenschnittstellen - Moduln mit Prozedur- und Daten-schnittstellen - Datenkapseln - Monitore
8. Zielrechnerbindung	- Für einen Zielrechner - Für eine Klasse von Zielrechnern - Cross - Systeme - ...
9. Aufbau des Software-entwicklungssystem	- Monolithische Systeme - Modulare Systeme
10. Architektur der Soft-wareentwicklungs-systeme	- Schichtenarchitektur - ...
11. Unterstützte Lebens-zyklusphasen	- Aufzählung der unterstützten Lebenszyklusphasen
12. Qualitätskontrolle	- Phasenorientierte Qualitäts-kontrolle - ...
13. Managementunterstützung	- Phasenorientiert - Phasenübergreifend

6.2. Technologische Werkzeuge im SIOS 1526

Entwicklungsbibliotheken:

- Quelltextbibliothek, allgemein)
- Quelltextbibliothek, Progr.-Sprache) SL

- Textbibliothek für Textbausteine)
- Dokumentationsbibliothek) TEXT-Dateien

- Modulbibliothek RL

- Phasenbibliothek PL

Methodische Anleitung zur Arbeit mit der Quelltextbibliothek:

1. Initialisieren der Diskette 5,25" INIT
   Sektor = 128, VOL1 = ROBINI, HDR1 = DATA
   BL = 80, BOE = 01001, EOE = 37024

2. Datei für Bibliothek vorbereiten FGEN
   VOL1 = LG5700, HDR1 = delete

3. Bibliothek installieren LBSV
   LW 5 für Bibliotheksdiskette zuweisen
   TYP = SL, LBTYP = S

4. Bücher eingeben, ändern, löschen EDIT
   Wenn erforderlich, dann Drucker zuordnen
   Diskette zuordnen "LG5700,5" (VOL/LW)
   Buchaufruf XXXXXX,99 (Buchname/,/Typ/Kdo)
   Typ 1 = Textbuch
   Typ 8 = Makrobuch

Methodische Anleitung zur Arbeit mit Textdateien

1. Initialisieren der Diskette 5,25" INIT

2. Datei zur Textverarbeitung einrichten TEXT
   LW 4 muß Textdatei enthalten
   Eingabe: MEIER11

INGE - Eingabeprogrammgenerator

(INGE, INTER, INGP, INGT, INGL)

- je Belegart ein Programm generieren
- Erstaufbau und Pflege von Stammdateien

INGE: - dialogorientiertes Erfassen der Belegparameter
- Übertragen in die PL

INTER: - Interpreter als Programmrahmen

INGP: - Druckprogramm als Protokoll in Tabellenform, auch teilweise möglich

INGT: - Test des Programms
- Schrittweise Abarbeitung
- Abschnittsweise Abarbeitung
- Register und Satzinhalt kann verfolgt werden

INGL: - Ausdrucken einer leeren Parameterliste für INGP

Datenträger: Diskette 8", 5,25", Kassette

Dateiform: eine Datei auf einem Datenträger
eine Datei auf mehreren Disketten
mehrere Dateien auf einer Diskette
geblockt/ungeblockt für Diskettendatei

Satzlänge: fest
Diskette: max 1024 Byte
Kassette: max 256 Byte
innere Satzstruktur beliebig

Ausgabe: Anfügen an den letzten Satz der Datei
= sequentielle Arbeitsweise

LIST - Generator (LISG)

- max. Satzlänge : 256 Byte (konstanter Satzaufbau)

- max. Blocklänge : 512 Byte

- Disketten : beliebige Sektorgröße

- Druck von Multivolume - Dateien

- Listenbreite: 59 Pos. = 150 mm Breite
85 Pos. = 210 mm Breite
128 Pos. = 300 mm Breite
180 Pos. = 460 mm Breite

- Auswahl von 20 Worten des Datensatzes zur Einbeziehung in Druck/Berechnung

- max. 3 Konstante (5-stellig)

- Bildung von 20 Worten aus arithmetischen Operationen je Zeile (Ergebnis, Konstante, Quellworte)

- max. 25 Summen

- Bildung von 25 Worten je Zeile, die mit einer beliebigen Maske (Dezimalpunkt) versehen werden

- Spaltensumme drucken, wenn
  . Seitenwechsel
  . Nummernwechsel eines Quellwortes
  . Dateiende

- Definition des Listenkopfes
  . Betriebsname/Oberbegriff (40 Zeichen)
  . Dateiname/Listenbezeichnung (32 Zeichen)
  . Überschrift 1 Zeichen = Listenbreite
  . Überschrift 2 Zeichen = Listenbreite
  . Überschrift 3 Zeichen = Listenbreite
  . Überschrift 4 Zeichen = Listenbreite

Dienstprogramme

EDIT für Quelltextgestaltung:

- Erfassen
- Einfügen, Anfügen
- Korrekturen
- Mischen
- Löschen
- Anzeige/Druck
- Verzeichnisanzeige

Bildschirmanzeige
obere 10 Zeilen Inhalt = Quelltext und Kommandoangebot
untere 6 Zeilen Dialogfunktionen

TEXT zum Dokumentieren:

- Erfassen
- Korrektur
- Druck
- Verzeichnisanzeige/-druck
- Reorganisieren

INIT zum Einrichten neuer Disketten:

- nur 2 defekte Spuren zulässig
- Spur 00 muß fehlerfrei sein

LBSV zur Bibliotheksverwaltung:

- Eröffnen einer Bibliothek
- Löschen
- Druck
- Umbenennen
- Kopieren
- Pflegen (Erweitern)

6.3. PSEUDO - ein dialogorientiertes Werkzeug

Anwendungskurzbeschreibung:
Pseudocode ist ein universelles, formatisiertes Mittel zur Beschreibung von Programmstrukturen für eine sprachlich gestützte Software-Entwicklung. Er dient dem Übergang von der natürlichen Umgangssprache zur formalen Programmiersprache, indem er vereinbarte Formalteile mit Sprachteilen verbindet.

I Formalteil	I Sprachteil I

PSEUDO/BC
Software-Werkzeug zur maschinellen Pseudocodeaufbereitung und -verarbeitung

Das Werkzeug PSEUDO unterstützt:
- den strukturierten Software-Entwurf
- die entwicklungsbegleitende Dokumentation
- den rationellen Übergang zum Implementieren
- das Minimieren des Zeitaufwandes

Die Hauptfunktionen sind:
- Pseudocodeaufbereitung mit
  Logikprüfung und Strukturdarstellung
- Pseudocodeverarbeitung mit
  Codegerüstgenerierung im PASCAL-1520
  (andere Programmiersprachen sind möglich)

PSEUDO-BC erfordert:
Hardware
- Bürocomputer A 5110, A 5120, A 5130
- Drucker
- Bildschirm und Tastatur- mindestens ein Diskettenlaufwerk

Software
- Betriebssystem SIOS 1526, BROS
- PASCAL-1520-System (ab Version 1.6)
- Phasen der Programme PCAUF und CGPAS
- Dienstprogramme

PSEUDO/BC
Problem- und Anwendungsdokumentation 1984:
Ingenieurhochschule Dresden
Sektion Informationsverarbeitung
8016 Dresden
Hans-Grundig-Str. 25

Grundform des Pseudocodes - PCM/1

Der Formalteil besteht aus Schlüsselworten, der Sprachteil enthält keine Schlüsselworte.

Pseudocode-Quelltext

Schlüsselwort (SW)	Nichtschlüsselwort (NSW)
Länge: max. 30 Zeichen	Länge: mehrere Zeilen
Standardzeichenfolge	Alle Zeichen der alphanumerischen Tastatur

	Eröffnen	Untersetzen	Abschließen
Block	START - HP		
	START - UP	DATA	
	START - BL	ACTION	STOPP
Grund-struktur	SEQ		ENDSEQ
	WHILE		ENDWHILE
	UNTIL		ENDUNTIL
	FOR		ENDFOR
	SELECT	CASE	
		OTHER	ENDSEL
Verweis	CALL		
	INCLUDE		
	LEAVE		

Ein Wort ist eine beidseitig mit Leerzeichen begrenzte Zeichenfolge.
PASCAL - 1520 - Anweisungen sind mit '%' zu kennzeichnen.

Der Pseudocode PCM/1 entspricht dem Pseudocode zur Daten- und Aktionsbeschreibung
/Handbuch der POS-Technologie /VEB ROBOTRON ZFZ /1980

Eingabeform 1

PC-Quelltext hat die Form:
- Schlüsselwort und Sprachteil belegen eine Zeile
  oder
- Schlüsselwort belegt eine Zeile und
  Sprachteil belegt eine Zeile

Eingabe: PC-Quelltext wird in unstrukturierter Form ab Zeilenanfang eingetastet.

Kommentare, welche im PASCAL-Quelltext erscheinen sollen, sind mit '#' am Zeilenanfang zu kennzeichnen.

Bezeichner — darf nur aus einem Wort bestehen, muß ungeteilt in einer Zeile stehen, darf mit '_' mehrere Teile verbinden.

Schlüsselwort — muß einheitlich mit kleinen oder großen Buchstaben eingegeben werden.

Innerhalb von Klammern müssen als Zeichenkettenkonstante verwendete Wörter
in Hochkomma eingeschlossen werden,
und ungeteilt in einer Zeile stehen,
innerhalb vorhandene Hochkomma müssen doppelt angegeben werden.

Eingabeform 2

PC-Quelltext ist kürzer oder länger als die Eingabezeile.

Kürzere Zeilen können in einer Zeile nacheinander eingetastet werden, wenn Bezeichner, Schlüsselworte und Klammerbereiche ungeteilt bleiben, z.B.:

```
ACTION  #AKTIONSBESCHREIBUNG  IF  A=B THEN
```

Längere Teile dürfen nur im Sprachteil mit Fortsetzungszeilen eingegeben werden, z.B.:

```
Funktion:  Transformation unstrukturierter Pseudocode-Beschrei-
  bung in eine strukturierte Darstellung
```

Kommentar, der im PASCAL-Quelltext stehen soll und mit '#' gekennzeichnet ist, darf mehrere Zeilen lang sein.

Bedienanleitung

1. BC einschalten, Betriebssystem laden.
   Empfehlung: Phasenbibliothek sollte alle notwendigen Dienstprogramme, das Programmiersprachensystem und die Phasen PCAUF, CGP.. enthalten.

2. Einrichten der Quelltextbibliothek für PC-Quelltext und PS-Quelltext.

3. Eingabe des PC-Quelltextes mit Kontrolldruck, evtl. Korrektur

4. Start der PC-Aufbereitung (PCAUF)
   Beachte: Bei Fehler erfolgt Programmabbruch mit Fehlermeldung, dann ab 3. Schritt weiterarbeiten.
   Fehlerfreies PC ist erreicht bei der Mitteilung:
   *** ENDE PROGRAMM - PCAUF ***

5. Ausgabe des strukturierten PC über den Drucker

6. Start der Codegerüstgenerierung (CGP..)
   Fehlerfreies Codegerüst ist erreicht bei der Mitteilung:
   *** ENDE PROGRAMM - CGP.. ***

7. Ausgabe des Codegerüstes über den Drucker.

8. Kopieren des Codegerüstes in eine andere Quelltextbibliothek zur Weiterarbeit.

Beispiel

```
0852 08.08.84
0853 start-UP swsieben
0854 action
0855 Funktion:Bei erkanntem Schlüsselwort der Länge 7 wird die
0858 Generierung für das Schlüsselwort: 'INCLUDE' realisiert.
0861 seq if (wort(1)='I') then seq
0863 Einlesen Name in Ausgabepuffer
0867 Übertragen '#MODUL: ' in Ausgabepuffer
0869 call PWRIT ENDSEQ ELSE
0870 SEQ setzen Fehlercode 307 call FEHLER
0872 ENDSEQ END IF
0873 ENDSEQ STOPP
```

Nach dem Aufbereiten mit PCAUF

Pseudocode SSW7

```
2801 26.09.84
2802  08.08.84
2803 #START-UP ***swsieben ***
2805 #ACTION
2806  Funktion:Bei erkanntem Schlüsselwort der Länge 7 wird die
2809  Generierung für das Schlüsselwort: 'INCLUDE' realisiert.
2812 #SEQ
2813 . #IF
2814 . .(WORT(1)='I')
2815 . . #THEN
2816 . .    #SEQ
2817 . .    .  Einlesen Name des einzufügenden Programmteils
2820 . .    .  Übertragen '#MODUL: ' in Ausgabepuffer
2822 . .    .  Übertragen NAME in Ausgabepuffer
2824 . .    . #CALL PWRIT
2825 . .    #ENDSEQ
2826 . . #ELSE
2827 . .    #SEQ
2828 . .    .  setzen Fehlercode 307
2830 . .    . #CALL FEHLER
2031 . .    #ENDSEQ
2832 . #ENDIF
2833 #ENDSEQ
2834 #STOPP
2835
```

6.4. ESEM - Softwareentwicklungswerkzeug (Entscheidungstabellenorientiertes Software-Entwicklungssystem

Grundgedanke:
Ein effektives Werkzeug zur Konstruktion von Steuermoduln und Verarbeitungsmoduln ist die Entscheidungstabellentechnik.
ESEM unterstützt sowohl POS-Technologie als auch ARS-Technologie und fördert das schnelle Generieren testfähiger Moduln im Dialog mit dem Computer.

Notation:
Das verwendete ET-Konzept enthält ET mit horizontal angeordneten Regeln (Regelzeilen).

- Eingabeform 1
  Eine ET-Zeile ist gleich/kleiner einer Eingabezeile mit max. 90 Zeichen.
  Beispiel:

```
I ETnn#B1#B2##A1#A2#GOTO    I     Kopfzeile
I nn#-#=A+1##=Z#=AZ+1#Rnn   I     Regelzeile
I TE                        I     Endezeile
```

- Eingabeform 2
  Sind in einer ET-Zeile mehr als 90 Zeichen erforderlich, wird in der Folgezeile die Zeichenkette ohne Angabe der Regelnummer fortgesetzt.

Leistungsparameter:
Assemblerteile sind unbegrenzt und dürfen zwischen den Entscheidungstabellen stehen.
Ein ET-Programm darf max. 99 ET enthalten.
Eine Entscheidungstabelle darf enthalten:
  max. 19 Bedingungsspalten
  max. 19 Aktionsspalten
  max. 19 Regelzeilen
  max. 1 Verbundaktionsspalte.
Die ELSE-Regel wird automatisch ergänzt, wenn keine vorhanden ist. Ihre Aktion ist dann stets "ENDE".

ESEM - Notationsvorschrift

Tabellenidentifikator := ET01...ET99
In der Kopfzeile dürfen nur Variable und GOTO stehen.
Regelnummer := 01...99 Lücken erlaubt!
Irrelevant-Anzeiger in der Regelzeile := "-" oder " 'Ziffer",
wenn mehrere Anzeiger übergangen werden sollen.

```
GOTO-Anzeiger := R01...R99  Regelnummer
              := E01...E99  ET-Identifikator
              := name       Marke außerhalb der ET
```

Spaltentrennzeichen := "#"

zwischen Bedingungs- und Aktionsteil "##"

Es sind nur Großbuchstaben, Ziffern und vereinbarte Sonderzeichen zugelassen.

```
Konstante := logisch  (0 = false, 1 = true)
          := integer  (0...32767)
          := real     (0...64537)
          Größere Werte können durch Multiplikation erzeugt
          werden.
```

Variable vom Typ (BOOL, INTEGER, REAL) werden bezeichnet mit Axxx, für Feldelemente sind indizierte Variable zu vereinbaren.

```
Operatoren := logisch (NOT, OR, AND, XOR)
           := arithmetisch (+, -, *, /, **, .MOD., .ABS.)
           := vergleichend (alle Vergleichsmöglichkeiten)
```

Vergleichsausführung wird vom Typ der Variablen bestimmt, die am Anfang des Ausdruckes steht
(B=B, I=I, R=R, R=I)

Marken der Form "E9999" werden vom ET-Übersetzer vergeben und sollten im Programm an anderer Stelle nicht benutzt werden.

Programmrahmen:
Beispiel

```
.AA.
PN __
ORG ____H
.AE.
.VA.
INTGER B1,B2,A1,A2...
.VE.
.AA.
LD SP, ____H
.AE.
ET__......
.SE.  logisches Ende
.#H.  physisches Ende
```

6.5. SEP am ESS A 5310 mit TEXT20

Einsatzgebiete:	- Pflichtenheft - Analysephase (Umgangssprache, Tabellen, Formalisierte Notationen) - Erstellen der Dokumentation (Anwenderbeschreibung, Bedienanleitung, Organisationshandbuch - Nutzer)
Textdateien: (ca. 36 Seiten auf Minidisketten)	- Entwurfsbibliothek je Mitarbeiter - Produktbibliothek je Auftrag/ Thema
Bausteindateien: (max. 17 Zeilen)	- Datentabellenköpfe - Dateibeschreibungsvorlage - Entscheidungtabellenvorlage - Strukturbeschreibungsvorlagen
Anschriftendateien:	- Zuordnen von Überschriften zu konstanten Texten - Einfügen von gelisteten Textteilen in vorgegebene Rahmen
Leistung:	- Erstellen von Druckvorlagen - Erstellen von Umdruckorginalen - Verwalten der Entwicklerdokumentation - Erstellen von Dokumentationsteilen entsprechend gegebener Anforderungen - Erfassen auf Diskette (nur Schreibmaschinen-Zeichensatz)